**IoT-Based Water Management System Using ESP32**

This project is a **smart water management system** that automatically controls a **water pump** and **a tap valve** based on water levels and flow rate. Additionally, it provides **remote manual control** through a **web interface** using the ESP32's built-in WiFi capabilities.

**Features**

✅ **Automatic Water Pump Control** - Turns ON/OFF the motor based on the tank water level.  
✅ **Automatic Tap Overflow Prevention** - Closes the valve if excessive water flow is detected.  
✅ **Web Interface for Remote Control** - Users can manually turn the motor ON/OFF via a webpage.  
✅ **Real-Time Monitoring** - Displays **water tank level** and **tap water flow rate** on the web interface.

**🛠 Components Used**

| **Component** | **Purpose** |
| --- | --- |
| **ESP32** | The main microcontroller, handles everything. |
| **Ultrasonic Sensor (HC-SR04)** | Measures the water level in the tank. |
| **Flow Sensor (YF-S201 or similar)** | Detects water flow rate at the tap. |
| **Relay Module (2-Channel)** | Controls the motor and the valve. |
| **Water Pump** | Fills the tank when needed. |
| **Solenoid Valve** | Closes the tap when overflow is detected. |

**Circuit Connections**

| **Component** | **ESP32 Pin** |
| --- | --- |
| **Ultrasonic Sensor** |  |
| - Trig Pin | **GPIO 5** |
| - Echo Pin | **GPIO 18** |
| **Flow Sensor** | **GPIO 19** |
| **Relay (Motor)** | **GPIO 23** |
| **Relay (Valve)** | **GPIO 22** |

**Power Connections**

* **ESP32** runs on **3.3V**.
* **Ultrasonic Sensor** and **Flow Sensor** need **5V**.
* **Relays, Pump, and Valve** use **external 5V or 12V (as per requirement).**

**Code Explanation**

**1️. Setting Up the Web Server & WiFi**

The ESP32 connects to WiFi and hosts a **web server** on port 80.

cpp

CopyEdit

WebServer server(80);

const char\* ssid = "Your\_WiFi\_SSID";

const char\* password = "Your\_WiFi\_Password";

The web interface allows users to **turn the motor ON/OFF manually**.

**2️. Water Level Measurement**

The **HC-SR04 ultrasonic sensor** is used to measure the **tank water level**.

cpp

CopyEdit

long getTankLevel() {

digitalWrite(TRIG\_TANK, LOW);

delayMicroseconds(2);

digitalWrite(TRIG\_TANK, HIGH);

delayMicroseconds(10);

digitalWrite(TRIG\_TANK, LOW);

return pulseIn(ECHO\_TANK, HIGH) \* 0.034 / 2;

}

💡 **Formula**: Distance = (Time × Speed of Sound) ÷ 2

**3️. Water Flow Detection**

The **flow sensor** generates pulses based on water flow. An **interrupt function** counts these pulses and calculates flow rate.

cpp

CopyEdit

void IRAM\_ATTR flowSensorISR() {

flowPulseCount++;

}

Every **second**, the flow rate is updated:

cpp

CopyEdit

flowRate = (flowPulseCount / 7.5); // Adjust for sensor calibration

flowPulseCount = 0;

**4️. Automatic Motor & Valve Control**

* **Motor turns ON** when water level **exceeds 40 cm**.
* **Motor turns OFF** when water level **drops below 10 cm**.

cpp

CopyEdit

if (tankLevel > 40) {

digitalWrite(RELAY\_MOTOR, LOW);

} else if (tankLevel < 10) {

digitalWrite(RELAY\_MOTOR, HIGH);

}

* If **water flow exceeds 10 L/min**, the **valve closes** to prevent overflow.

cpp

CopyEdit

if (flowRate > 10) {

digitalWrite(RELAY\_VALVE, LOW);

} else {

digitalWrite(RELAY\_VALVE, HIGH);

}

**5️. Web Interface for Manual Control**

The ESP32 serves a **basic webpage** where users can **turn ON/OFF the motor**.

cpp

CopyEdit

void handleRoot() {

String html = "<html><body><h2>Water Management System</h2>";

html += "<p><a href='/motor/on'>Turn ON Motor</a></p>";

html += "<p><a href='/motor/off'>Turn OFF Motor</a></p>";

html += "<p>Water Tank Level: " + String(getTankLevel()) + " cm</p>";

html += "<p>Tap Water Flow Rate: " + String(flowRate) + " L/min</p>";

html += "</body></html>";

server.send(200, "text/html", html);

}

🚀 When a user clicks **Turn ON Motor**, this function runs:

cpp

CopyEdit

void motorOn() {

digitalWrite(RELAY\_MOTOR, LOW);

server.send(200, "text/html", "<p>Motor Turned ON</p><a href='/'>Back</a>");

}

For **Turn OFF Motor**, this function runs:

cpp

CopyEdit

void motorOff() {

digitalWrite(RELAY\_MOTOR, HIGH);

server.send(200, "text/html", "<p>Motor Turned OFF</p><a href='/'>Back</a>");

}

This lets the user manually control the motor **unless the automatic logic overrides it**.

**Summary**

This **IoT-based water management system** ensures **efficient water usage** by **automating water pumps and preventing tap overflow**. The system also includes **manual control** via a **web-based interface**, allowing flexibility in operation.

🔹 **Automatic Control** ensures **no wastage of water**.  
🔹 **Web Interface** allows **remote monitoring & control**.  
🔹 **ESP32 with WiFi** makes it **cost-effective & scalable**.